UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/069,352	08/07/2002	Gilbert Wolrich	10559-308US1	7931

20985 7590 05/21/2008
FISH & RICHARDSON, PC
P.O. BOX 1022
MINNEAPOLIS, MN 55440-1022

EXAMINER	
HUISMAN, DAVID J	
ART UNIT	**PAPER NUMBER**
2183	
MAIL DATE	**DELIVERY MODE**
05/21/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/069,352
Filing Date: August 07, 2002
Appellant(s): WOLRICH ET AL.

Denis G. Maloney, Reg. No. 29,670
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed on March 10, 2008, appealing from the Office action mailed on October 11, 2007.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

- Bhattacharya, U.S. Patent No. 5,704,054 (December 30, 1997)
- Sproull et al., "The Counterflow Pipeline Processor Architecture," 1994, pp.48-59 (herein cited as extrinsic evidence)

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 102

1. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

2. Claims 1-2, 6, 8-9, 20-21, 25, and 27-28 are rejected under 35 U.S.C. 102(b) as being anticipated by Bhattacharya, U.S. Patent No. 5,704,054.

3. Referring to claim 1, Bhattacharya has taught a method of operating a multi-threaded processor comprising:

a) receiving data specified by execution of a fast-write instruction in one of multiple threads processing on the multi-threaded processor, the one of the multiple threads identified by a processing thread number. See column 6, lines 32-57. Threads, which are inherently identified by number (so that they may be distinguished), also include instructions which specify result data.

b) the fast-write instruction further specifying a register, the register having multiple groups of bits, each group of bits associated with a corresponding thread of the multiple threads processing on the multi-threaded processor. See Fig.6, component 42, and column 6, lines 32-41. Note that a result register, which holds results specified by a "fast-write" instruction, is divided into at least N groups for N threads. Each thread will then write to its portion of the register based on an address identifier.

c) selecting a group of bits associated with the one of the multiple threads, the group of bits being selected from the multiple groups of bits of the register specified by the fast-write instruction according to the processing thread number. Again, see column 6, lines 32-57. When a thread is to write to the result register specified by the fast write, the thread number is used to select the group of bits (portion) to which the result is written.

d) loading the data into the selected bit positions of the register. See column 6, lines 32-57. A thread writes a result to the result register (loads data into the result register).

4. Referring to claim 2, Bhattacharya has taught a method as described in claim 1. Bhattacharya has further taught that the register is a control and status register (CSR). See Fig.6, component 42, and note that a result register, when used as an operand of a dependent instruction, controls that instruction. In addition, it is a status register because writing a result implies completion status of the writing instruction.

5. Referring to claim 6, Bhattacharya has taught a method as described in claim 1. Bhattacharya has further taught that the processing thread represents processing in a micro engine of a multithreaded processor. See Fig.5, and note a pipeline (micro engine) processes a thread.

6. Referring to claim 8, Bhattacharya has taught a method as described in claim 1. Bhattacharya has further taught that the fast-write instruction comprises a token. See Fig.6 and note the opcode field (OP) in the instruction. This is a token in that this specifies that a particular execution unit is to take control during execution of that instruction.

7. Referring to claim 9, Bhattacharya has taught a method as described in claim 8. Bhattacharya has further taught that the token represents overriding qualifiers. Since the opcode

in the instruction contains multiple bits and also specifies that the current contents of a register are to be overwritten with the result of the instruction, the opcode bits may be considered overriding qualifiers, as these bits result in a new result overriding an old result.

8. Referring to claims 20-21, 25, and 27-28, claims 20-21, 25, and 27-28 are rejected for the same reasons set forth in the rejection of claims 1-2, 6, and 8-9, respectively, because Bhattacharya has taught instructions stored on a medium for performing the method of claims 1-2, 6, and 8-9.

Claim Rejections - 35 USC § 103

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claims 3-5, 10-11, 14, 17, 22-24, 29-30, 33, and 36 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bhattacharya.

11. Referring to claim 3, Bhattacharya has taught a method as described in claim 2. Bhattacharya has not explicitly taught that the control and status register is coupled to a 64-bit wide first-in first-out (FIFO) bus. However, as shown in *In re Rose, 105 USPQ 237 (CCPA 1955)*, changes in size are generally not given patentable weight or would have been an obvious improvement. Specifically, the size of Bhattacharya's bus is not disclosed, but a 64-bit bus is a common bus size, which allows for more data to be transported at once than on a 32-bit bus or 16-bit bus, for instance. Consequently, it would have been obvious to one of ordinary skill in the

art at the time of the invention to modify Bhattacharya's bus to be 64-bits wide. Furthermore, the bus is inherently FIFO because data that is sent over the bus at time A will arrive at its destination before data that is sent over the bus at time B (where B > A).

12. Referring to claim 4, Bhattacharya, as modified, has taught a method as described in claim 3. Furthermore, it is deemed inherent in Bhattacharya that the FIFO bus interfaces with Media Access Controller (MAC) devices. There must be devices that control the means for communicating across a bus (for instance, the buses of Fig.3). These devices are media access controllers as they control media access.

13. Referring to claim 5, Bhattacharya has taught a method as described in claim 1. Bhattacharya has not explicitly taught that the data represents hexadecimal mask values 0 to 0x3FF. However, the examiner asserts that results of a variety of operations may take on any number of values, including those in the range 0 to 0x3FF. The machine would simply have to be at least capable of storing 12-bit results, and storing larger results is well known in the art. A large result allows for the representation of more values. That is, clearly more values can be represented with 16 bits, for instance, than with 1 bit. Consequently, in order to increase the range of representation of data values, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Bhattacharya to include at least a 12-bit result for storing values in the range 0 to 0x3FF.

14. Referring to claim 10, Bhattacharya has taught a method as described in claim 9. Bhattacharya has not explicitly taught that the token is a 32-bit word. However, Bhattacharya has not disclosed the instruction size in any way, and the examiner asserts that 32-bit opcodes are well-known and have been used in the art. The larger the opcode, the more operations that can

be encoded. In addition, *__In re Rose__, 105 USPQ 237 (CCPA 1955)*, changes in size are generally not given patentable weight or would have been an obvious improvement. As a result, in order to encode many operations in Bhattacharya, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Bhattacharya's token (opcode) to be a 32-bit word.

15. Referring to claim 11, Bhattacharya, as modified, has taught a method as described in claim 10. Furthermore, Bhattacharya has taught that a token format comprises: an OV field in bit 31, a micro engine (UENG) ADDR field in bits 30:28, a reserved field in bits 27:16, an OV field in bit 15, a fast write data field in bits 14:5, a reserved field in bits 4:3, an OV field in bit 2, and a CTX field in bits 1:0. That is, given Bhattacharya's 32-bit token (resulting from the modification), it can be said that the bits of the token correspond to the above fields as they are merely names, labels, or descriptions of fields, that impart no functionality.

16. Referring to claim 14, Bhattacharya, as modified, has taught a method as described in claim 11. Bhattacharya has further taught that bits 27:16 return 0 when read. That is, in the case where bits 27:16 are set to 0, they will return 0 when read. Note that these bits may all be set to 0 because each bit can take on a 0 or 1 value.

17. Referring to claim 17, Bhattacharya, as modified, has taught a method as described in claim 11. Bhattacharya has further taught that bits 4:3 return 0 when read. That is, in the case where bits 4:3 are set to 0, they will return 0 when read. Note that these bits may all be set to 0 because each bit can take on a 0 or 1 value.

18. Referring to claims 22-24, 29-30, 33, and 36, claims 22-24, 29-30, 33, and 36 are rejected for the same reasons set forth in the rejection of claims 3-5, 10-11, 14, and 17, respectively,

because Bhattacharya has taught instructions stored on a medium for performing the method of claims 3-5, 10-11, 14, and 17.

(10) Response to Argument

Prior to addressing appellant's arguments, the examiner believes that the Board would benefit from a brief description of Bhattacharya, the prior art challenged by appellant. In Bhattacharya (and as known in the art), each instruction executed by the processor includes two source registers and a destination register (see Fig.6). That is, the instruction is of the exemplary form "OP R1, R2, R3" (where OP (opcode) specifies the operation to be performed using the data in specified registers (R1, R2, R3). For instance, OP could be "ADD" which would result in the processor adding the contents of source registers R2 and R3 and storing the result in destination register R1). Bhattacharya is built on the concept of a counterflow pipeline introduced by Sproull et al. (see column 2, lines 6-14, of Bhattacharya, and note throughout that Bhattacharya refers to the CFPP, which is the acronym for the counterflow pipeline processor). In the CFPP processor, each instruction has a "binding" for each register in the instruction. The binding includes the register name/address (i.e., R1), the value in the register, and a valid bit specifying whether the data in the register is valid. See the extrinsic evidence (Sproull), page 49, column 1, section "Basic Structure", paragraph 2. Therefore, the instruction (OP R1, R2, R3) would have a generalized format as follows:

OP - R2 - R2d - R2v - R3 - R3d - R3v - R1 - R1d - R1v

Such a format is made clear from Fig.1 and Fig.2 in Sproull. Note that a first binding would include a register name (R2), the data stored in the register named (R2d), and a valid bit associated with the data in the register named (R2v). One binding exists for each register specified by the instruction. The bindings are required for forwarding purposes, wherein forwarding is the well-known concept of supplying a subsequent instruction with a prior instruction's result. See the last paragraph beginning on page 49 of Sproull and continuing onto page 50. Essentially, when an instruction produces a result, it will be written to the register file (in register R1 in the above example) and the binding will then be sent through a result pipeline in the direction opposite the flow of the instruction pipeline. Then, when the result binding matches a source binding of another instruction, the data is transferred from the result binding to the instruction (forwarding) so that stalling will not occur, as is known in the art. Note page 51 of the extrinsic evidence for a detailed example on how this counterflow works.

The unique concept in Bhattacharya, and the concept of relevance with respect to appellant's claims, is that a CFPP processor is modified to realize a multithreaded CFPP processor such that instructions from multiple threads propagate through the pipeline. To accommodate multiple threads, Bhattacharya modifies the registers of Sproull's CFPP architecture such that each register has N sections, one section for each thread. Note the difference between Fig.2 (prior art, single-threaded CFFP) and Fig.6 (multithreaded CFFP) of Bhattacharya. Fig.2 merely shows an instruction containing one value per register 52, whereas Fig.6 shows N values per register, one for each thread. Hence, depending on the thread, the appropriate section of the register is written. See column 6, lines 32-41. As disclosed, if three threads were to exist, for instance, then a result register must include three separate sections

(separate groups of bits), each holding a result from a different thread. As an example, suppose the following three instructions from three different threads exist:

Thread 1: ADD R1, R2, R3
Thread 2: ADD R1, R4, R5
Thread 3: ADD R1, R6, R7

In Bhattacharya, if R2 holds a value of 10, R3 holds a value of 20, R4 holds a value of 30, R5 holds a value of 40, R6 holds a value of 50, and R7 holds a value of 60, then result register R1 would look as follows after execution of the three instructions:

R1:	110	70	30

Note that the ADD instruction of thread 1 selects the first (rightmost) group of bits and wrote the result to that group. In similar fashion, the other thread instructions selected their respective groups of bits for result writing. While this example is meant to be exemplary, one of ordinary skill in the art would recognize that thread 1 instructions could instead be configured to select the leftmost group of bits, for instance. Which group is selected is based on the design.

1) With respect to claim 1:

On pages 8-10 of the appeal brief, appellant argues, in substance, that:

> "Claim 1 is directed to a method of operating a multithreaded parallel processor. Appellant contends that Bhattacharya neither describes nor suggests at least the features of "selecting a group of bits associated with the one of the multiple threads, the group of bits being selected from...multiple groups of bits of [a] register specified by [a] fast-write instruction according to [a] processing thread number," as called for in claim 1...
>
> Bhattacharya neither describes nor suggests in these passages the feature of: "...the group of bits being selected from...multiple groups of bits of [a] register specified by [a] fast-write

instruction ...". Rather, in the relied on passages Bhattacharya explains that within a pipeline an instruction register and a results register are each provided with plural address positions, and that each register must contain at least enough registers to store instructions and data associated with each thread.

The relationship between an instruction and data is disclosed by Bhattacharya as follows:

> **Each of the source and destination registers includes a section 58 that carries a binding value (e.g. a register name in register file 12). In similar manner, result registers 60 and 62...contain binding value sections 64. By comparison of binding values 58 and 64 in logic circuit 44, a pipeline stage is able to determine which data passing through result register 42 is to be associated with an instruction in instruction register 40.**

That is, in Bhattacharya, the instructions in the source register and the data in the results register are mapped by binding values in a logic circuit. However, this is not what is called for in claim 1, which requires that "the fast-write instruction further specifying a register," as required by Appellant's independent claim 1. Because Bhattacharya neither discloses nor suggests at least the features of "selecting a group of bits associated with the one of the multiple threads, the group of bits being selected from...multiple groups of bits of [a] register specified by [a] fast-write instruction according to [a] processing thread number", Appellant's independent claim 1 is not anticipated by Bhattacharya."

This argument, while fully considered by the examiner, has been deemed non-persuasive.

It is noted by the examiner that appellant has cited Bhattacharya, column 3, lines 6-13, as the reason why Bhattacharya has not taught the claimed invention. The examiner disagrees that this is a valid reason because this passage has nothing to do with writing a result to a register in the manner claimed by appellant. This cited passage merely explains how forwarding is performed in Bhattacharya, as described above.

The examiner asserts that Bhattacharya is concerned with how a result is written to a register and how a result is forwarded to a subsequent instruction. These are mutually exclusive operations. The claims focus on the former concept of how a result is written to a register. Hence, this is the focus of the examiner in making his rejection. However, Appellant's arguments are related to the latter concept of how forwarding is performed and not how a result is written to a result register. The examiner asserts that this examination is concerned with how the result is written, and not what is done with the result after it is written, and consequently,

appellant's arguments are non-effective. Specifically, appellant argues that the claim is not anticipated because, in Bhattacharya, "the instructions in the source register and the data in the results register are mapped by binding values in a logic circuit." While the examiner asserts that this is true, this has nothing to do with selecting bits in a result register and writing a result to those bits.

The examiner feels that it is clearly stated, in column 6, lines 32-41, that a result register includes N result sections (groups of bits), one for each thread. And, when a particular instruction specifies a result (for instance, an ADD instruction would specify an addition result), that result is written to a particular section based on the thread that includes the instruction. Since there is a one-to-one correspondence between threads and sections of a result register, and each thread writes to a corresponding section, then the section of the register written is solely dependent on the identification of the thread. The examiner asserts that in order to distinguish between threads, they must each be uniquely identifiable, and therefore include thread numbers. As a side note, it should be further realized that there is no requirement in the claim that a physical thread number be stored anywhere within a processor. That is, there is nothing stopping a person from simply assigning thread numbers mentally and realizing that the processor selects a register section based on that mental assignment. For instance, if an individual knows that three threads will exist in the system, then, in addition to the processor inherently distinguishing between threads, that individual could number the threads 1, 2, and 3 (or T1, T2, and T3). And, if T1 writes to a first group of bits, then it can be said that the first group of bits is selected according to the processing thread number. However, despite this mental identification, the system of Bhattacharya must be able to distinguish (identify) each thread to determine group of

bits to select and write to in the result register. The mental identification explanation merely points out the broadness of the claim language.

As mentioned, the passage that appellant cites from Bhattacharya discusses how forwarding of a result is performed. And, this is not relevant to the claim, as appellant only claims writing a result to a result register. The passage of Bhattacharya merely points out that source register bindings of an instruction flowing through a pipeline in one direction are compared to result register bindings of results previously written flowing through a pipeline in an opposite direction such that when a match occurs, the result, may be passed to the instruction which needs it for operation. For example, if a first instruction "ADD R1, R2, R3" writes an addition result to R1, and then a number cycles later, a second instruction "MULT R4, R5, R1" is encountered, then the result in R1 will be passed (forwarded) to the multiplication instruction so that a multiply operation may occur. This is one example of a forwarding situation that is touched upon in the passage of Bhattacharya cited by appellant.

In summation, the examiner asserts that column 6, lines 32-41, of Bhattacharya clearly teach a result register having multiple groups of bits (one group for each thread), where each thread writes to a group. Hence, that group must be selected based on the thread identified. The fact that "in Bhattacharya, the instructions in the source register and the data in the results register are mapped by binding values in a logic circuit", as argued by appellant, does not mean that Bhattacharya does not anticipate the claimed invention. It merely means that the means exist to perform forwarding after a result is written. Consequently, the examiner respectfully requests that the Board affirm the examiner on this matter.

2) With respect to claim 2:

On pages 10-11 of the appeal brief, appellant argues, in substance, that:

> "Bhattacharya neither describes nor suggests at least the feature of "the register [being] a control and status register (CSR).
>
> The examiner contends that for
>
> > **...claim 2, Bhattacharya has taught a method as described in claim 1. Bhattacharya has further taught that the register is a control and status register (CSR). See Fig.6, component 42, and note that a result register, when used as an operand of a dependent instruction, controls that instruction. In addition, it is a status register because writing a result implies completion status of the writing instruction.**
>
> Further, the examiner contends that Bhattacharya's results register is a "control and status register." Appellant disagrees and contends that Bhattacharya's instruction would then be required to specify the register. Bhattacharya discloses that:
>
> > **[a] pipeline stage wherein an instruction operation code is to be executed is, for instance, adder launch module 32 wherein an add instruction is both recognized and causes data having a proper binding value to be fed to adder 28 wherein a sum is created and then fed to add recover module 34. The result data then propagates to register file 12 and is stored.**
>
> The instruction here merely causes data to be acted upon by some module before it is delivered to some register. Therefore, assuming arguendo that this register is a control and status register, claim 2 still distinguishes over Bhattacharya because the register, as disclosed by Bhattacharya is not specified by the instruction, as required by Appellant's claim 2. Therefore, Bhattacharya neither describes nor suggests at least the features of"...the fast-write instruction ... specifying a register...", and thus Appellant's claim 2 is not anticipated by Bhattacharya."

This argument, while fully considered by the examiner, has been deemed non-persuasive.

It appears that appellant doesn't have an issue with the examiner's explanation of how the register is a control and status register. Instead, appellant argues that even if it were a control and status register, it isn't specified by a fast-write instruction. The examiner asserts that, in Bhattacharya, an instruction specifies at least one register of a number of registers to store the result of the instruction. See column 2, lines 37-43, and column 3, lines 6-8, of Bhattacharya, and Sproull, page 49, column 1, section "Basic Structure", paragraph 2. Therefore, a result register is clearly specified by an instruction.

Since an instruction clearly specifies a result register, and a result register is a control and status register for the reasons described in the rejection of claim 2, the Board is respectfully requested to affirm the examiner on this matter.

3) With respect to claim 6:

On pages 11-12 of the appeal brief, appellant argues, in substance, that:

> "Bhattacharya neither describes nor suggests at least the features of 'the one of multiple threads is processed on a micro engine of a multi-threaded processor.'...
> Bhattacharya provides micro sequencers and not micro engines as required by appellant's claim 6."

This argument, while fully considered by the examiner, has been deemed non-persuasive.

The examiner has been unable to find an explicit definition of micro-engine in the specification that would distinguish it from any general component that simply processes threads in Bhattacharya. The examiner pointed out that the pipeline of Fig.5 is a micro-engine. It contains multiple program counters and micro-sequencers, and all the logic necessary to execute threads. Hence, this is a micro-engine.

The Board is respectfully requested to affirm the examiner on this matter.

4) With respect to the arguments related to the remaining rejected claims, the examiner's response would be similar to those responses set forth above.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/David J. Huisman/
Primary Examiner, Art Unit 2183
May 8, 2008

Conferees:

/Eddie P Chan/
Supervisory Patent Examiner, Art Unit 2183

/Kevin L Ellis/
Acting SPE of Art Unit 2187